A Branch-and-Cut Algorithm for Equitable Coloring based on a Formulation by Representatives

Laura Bahiense[1] Yuri Frota[2] Nelson Maculan[3]
Thiago F. Noronha[4] Celso C. Ribeiro[5]

1 Introduction and motivation

Let $G = (V, E)$ be an undirected graph, where V is the set of vertices and E is that of edges. An equitable k-coloring of G is a partition of V into k disjoint stable subsets such that the difference on the cardinalities of any two subsets is at most one. Each subset is associated with a color and called a *color set*. The Equitable Coloring Problem (ECP) consists of finding the minimum value of k such that there is an equitable k-coloring of G. This number is said to be the *equitable chromatic number* of G and it is denoted by $\chi_=(G)$.

The equitable coloring problem was first introduced in [7], motivated by an application to municipal garbage collection [9]. It was proved to be NP-hard in [5]. A branch-and-cut algorithm based on an integer programming

[1] Programa de Engenharia de Produção, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ 21941-972, Brazil, laura@pep.ufrj.br.
[2] Instituto de Computação, Universidade Estadual de Campinas, Campinas, SP 13083-970, Brazil, abitbol@ic.unicamp.br.
[3] Programa de Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ 21941-972, Brazil, maculan@cos.ufrj.br.
[4] Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ 22453-900, Brazil, tfn@inf.puc-rio.br.
[5] Departamento de Ciência da Computação, Universidade Federal Fluminense, Rio de Janeiro, RJ 24210-240, Brazil, celso@ic.uff.br.

formulation and Gomory and lifted cover cuts was proposed in [1]. A polyhedral approach for the equitable coloring problem was proposed in [6], but the largest randomly generated instances solved by this approach had only 35 nodes.

This paper presents a new exact algorithm for the problem, based on a generalization of the 0-1 formulation by representatives for the vertex coloring problem [2]. A tabu search heuristic provides upper bounds, and a cutting plane procedure is used to strengthen the lower bounds. These two methods, together with a branching strategy, are used in a branch-and-cut algorithm for solving ECP. Section 2 summarizes the proposed branch-and-cut algorithm. Computational experiments are reported in Section 3. Concluding remarks are drawn in the last section.

2 Branch-and-cut algorithm

In the formulation by representatives, we choose one vertex to be the representative of all vertices colored with the same color. Therefore, each vertex is in one of the following two states: either it represents its color or there exists another vertex that represents its color. To break symmetries, we establish that a vertex $u \in V$ can only represent the color of a vertex $v \in V$ if $u < v$. Therefore, the representative vertex of a color is that with the smallest index among all those with this same color. The representative formulation has been successfully applied to other graph coloring problems [2,4].

Let $A(u) = \{v \in V : (u, v) \notin E, v \neq u\}$ be the *anti-neighborhood* of a vertex $u \in V$ (i.e., the subset of vertices that are not adjacent to u). Furthermore, let $A_>(u) = \{v \in A(u) : v > u\}$ be the *out-anti-neighborhood* of a vertex $u \in V$ (i.e., the vertices that cannot represent vertex u) and $A_<(u) = \{v \in A(u) : v < u\}$ be the *in-anti-neighborhood* of vertex $u \in V$ (i.e., the vertices that may represent vertex u, itself excluded). We also define $A'_>(u) = A_>(u) \cup \{u\}$ and $A'_<(u) = A_<(u) \cup \{u\}$, for any $u \in V$. Given a subset of vertices $V' \subseteq V$, we denote by $E[V']$ the subset of edges induced in the graph $G = (V, E)$ by V'. A vertex $v \in A_>(u)$ is said to be *isolated* in $A_>(u)$ if $E[A_>(u)] = E[A_>(u) \setminus \{v\}]$ (i.e., vertex v has no adjacent vertex in $A_>(u)$).

We define the binary variables x_{uv} for all $u \in V$ and for all $v \in A'_>(u)$, such that $x_{uv} = 1$ if and only if vertex u represents the color of vertex v; otherwise $x_{uv} = 0$. Let L_w and U_w be, respectively, integer lower and upper bounds for the value w of the cardinality of the largest color set in an equitable coloring of G (i.e., the cardinality of each color set is either w or $w-1$). We introduce

the equilibrium variable $y_i \in \{0,1\}$, for every integer $i \in [L_w, U_w]$, such that $y_i = 1$ if the cardinality of the largest color set of the equitable coloring of G is i; $y_i = 0$ otherwise. We also introduce variables $z_{ui} \in \mathbb{R}$, where $z_{ui} = x_{uu} \cdot y_i$ at any integer solution of ECP, for every $u \in V$ and every integer $i \in [L_w, U_w]$. For sake of conciseness, we omit the details of the computations of L_w and U_w.

We also define $V^s = \{u \in V : A_<(u) = \emptyset\}$ as the set of vertices whose in-anti-neighborhoods in G are empty (i.e., the set of vertices that are always representatives). Since vertices in V^s are always representatives, then $x_{uu} = 1$ in any feasible solution, for any $u \in V^s$. Therefore, these variables may be removed from the formulation. ECP can be formulated as the following integer programming problem. For sake of simplicity, we use the notation $\beta_u = 1$ if $u \in V^s$; otherwise $\beta_u = x_{uu}$. The integer programming problem LF_2 described below is said to be the *formulation by representatives* of the equitable coloring problem:

$$\min \sum_{v \in V \setminus V^s} x_{vv} + \mid V^s \mid \tag{1}$$

subject to:

$$\sum_{v \in A'_<(u)} x_{vu} = 1, \qquad \forall u \in V \setminus V^s \tag{2}$$

$$x_{uv} + x_{uw} \leq \beta_u, \qquad \forall u \in V, \forall (v,w) \in E : v, w \in A_>(u) \tag{3}$$

$$x_{uv} \leq x_{uu}, \qquad \forall u \in V \setminus V^s, \forall v \in A_>(u) : v \text{ is isolated in } A_>(u) \tag{4}$$

$$\beta_u + \sum_{v \in A_>(u)} x_{uv} \leq \sum_{i=L_w}^{U_w} i \cdot z_{ui}, \qquad \forall u \in V \tag{5}$$

$$2 \cdot \beta_u + \sum_{v \in A_>(u)} x_{uv} \geq \sum_{i=L_w}^{U_w} i \cdot z_{ui}, \qquad \forall u \in V \tag{6}$$

$$\sum_{i^=L_w}^{U_w} y_i = 1 \tag{7}$$

$$z_{ui} \leq y_i, \quad z_{ui} \leq \beta_u, \quad z_{ui} \geq y_i + \beta_u - 1, \quad \forall u \in V, \forall i \in [L_w, U_w] \tag{8}$$

$$x_{uv} \in \{0,1\}, \qquad \forall u \in V, \forall v \in A'_>(u) \tag{9}$$

$$y_i \in \{0,1\}, \qquad \forall i \in [L_w, U_w] \tag{10}$$

$$z_{ui} \in \mathbb{R}, \qquad \forall u \in V, \forall i \in [L_w, U_w]. \tag{11}$$

The objective function (1) counts the number of representative vertices, i.e., the number of colors. Constraints (2) enforce that each vertex $u \in V \setminus V^s$

must be represented either by itself or by another vertex v in its in-anti-neighborhood. Inequalities (3) enforce that adjacent vertices have distinct representatives. Inequalities (3) together with constraints (4) ensure that a vertex can only be represented by a representative vertex. Inequalities (5)-(8) guarantee that the difference on the cardinalities of any two color sets is at most one.

The branching strategy plays a major role in the success of a branch-and-cut algorithm. Branching on the x_{uv} variables, with $u \in V$ and $v \in A'_{>}(u)$, is not efficient because most of them are null in integer solutions. Therefore, our branching strategy is based on the cardinality variables y_i. Branching on the x_{uv} variables starts only after all the y_i variables are integer.

To improve the linear relaxation bound, we use the two families of valid inequalities described in detail in [3]: *external cuts* and *internal cuts*.

The adaptive tabu search heuristic of Touhami [8] that provides upper bounds for the frequency assignment problem was adapted to give initial feasible solutions for ECP at each node of the branch-and-cut tree. Due to the limitation of space, we omit here the details of this algorithm.

3 Computational experiments

Algorithm B&C-LF_2 (based on formulation LF_2) and the branch-and-cut in [1] were implemented in C++ and compiled with version v3.41 of the Linux/GNU compiler. XPRESS version 2005-a was used as the linear programming solver. All experiments were performed on an AMD-Atlon machine with a 1.8 GHz clock and one Gbyte of RAM memory. The graphs used in the experiments have been randomly generated exactly as described in [6].

The branch-and-cut algorithm in [6] was able to solve only very small ECP instances with at most 35 nodes. Due to the limitation of space, and since our approach was able to solve much larger instances, we report only the results obtained by algorithm B&C-LF_2 and the branch-and-cut in [1] for graphs with 60 nodes and edge densities ranging from 10% to 90%. Each ale_n_p_s instance has n vertices and an uniform probability p that any two vertices share an edge. The last digit s is an instance differentiator.

The first three columns in Table 1 display the name of each instance, its number of vertices, and its number of edges. The next four columns give the lower bound, the upper bound, the number of evaluated nodes in the branch-and-cut tree, and the CPU times (in seconds) for finding the optimal solution provided by the algorithm in [1]. The last four columns give the same results for algorithm B&C-LF_2. A missing time entry in this table indicates that the

			B&C [1]				B&C-LF_2			
Graph	$\mid V \mid$	$\mid E \mid$	LB	UB	nodes	time	LB	UB	nodes	time
ale60_10_1	60	166	4	4	35	6	4	4	20	38
ale60_10_2	60	148	4	4	29	6	4	4	11	15
ale60_10_3	60	150	4	4	13	5	4	4	8	16
ale60_10_4	60	173	4	4	25	8	4	4	15	23
ale60_10_5	60	152	4	4	57	9	4	4	9	14
ale60_30_1	60	529	6	7	82426	-	7	8	4368	-
ale60_30_2	60	519	6	7	76495	-	7	8	3945	-
ale60_30_3	60	535	6	7	152382	-	7	7	41	61
ale60_30_4	60	509	6	7	206105	-	7	7	137	111
ale60_30_5	60	506	6	7	223443	-	7	8	3873	-
ale60_50_1	60	880	8	12	8112	-	11	11	351	214
ale60_50_2	60	853	8	11	5178	-	10	11	13501	-
ale60_50_3	60	858	8	13	38413	-	10	10	5	15
ale60_50_4	60	849	9	13	14105	-	10	11	36708	-
ale60_50_5	60	903	8	13	5941	-	11	11	657	388
ale60_70_1	60	1239	12	25	4826	-	16	16	33	19
ale60_70_2	60	1240	12	18	3401	-	16	16	61	28
ale60_70_3	60	1240	13	20	5966	-	16	16	771	252
ale60_70_4	60	1209	12	17	6153	-	16	16	63	28
ale60_70_5	60	1261	12	22	2320	-	16	16	71	28
ale60_90_1	60	1559	22	25	3823	-	24	24	1	9
ale60_90_2	60	1561	18	31	456	-	31	31	1	9
ale60_90_3	60	1571	19	26	858	-	25	25	1	9
ale60_90_4	60	1583	22	37	4081	-	25	25	2	9
ale60_90_5	60	1606	23	38	1699	-	26	26	1	9

Table 1
Computational results for randomly generated graphs with 60 nodes.

problem could not be solved within two hours of processing time. In this case, we display the lower bound, the upper bound, and the number of evaluated nodes at the time the algorithm was stopped.

The lower bounds provided by formulation LF_2 were always better than or equal to those provided by the branch-and-cut algorithm in [1]. The branch-and-cut algorithm B&C-LF_2 solved 20 out of the 25 instances in Table 1 within two hours of processing time, while the algorithm in [1] managed to solve only five out of the 25 instances with edge density 10%.

The average relative gap $(UB - LB)/LB$ observed for the branch-and-cut algorithm B&C-LF_2 over the 25 instances in Table 1 was only 2.5%, while the same gap for the algorithm in [1] was 37.3%. Algorithm B&C-LF_2 achieves its best performance on instances with edge densities larger than 50%. This is due to the fact that the larger is the graph density, the larger are the sizes of the cliques, odd holes, and odd anti-holes used to generate the internal and external cuts. The hardest instances for B&C-LF_2 were those with edge densities ranging from 30% to 50%, because the cliques, odd holes, and odd anti-holes found in such instances were not great enough to generate good internal and external cuts. However, the largest absolute gap $UB - LB$ observed for algorithm B&C-LF_2 on the 25 instances in Table 1 was of only one color,

while the same value for the branch-and-cut algorithm in [1] was as large as 15 colors (for instance ale60_90_4).

4 Concluding remarks

We proposed a branch-and-cut algorithm for the equitable graph coloring problem, based on its formulation by representatives. Instances with up to 60 nodes and edge densities ranging from 10% to 90% have been solved to optimality, while the largest problems solved to date in [6] had only 35 nodes. The proposed branch-and-cut algorithm B&C-LF_2 clearly outperformed those in [1,6].

References

[1] Bahiense, L., S. Jurkiewicz, A. Lozano, M. Pimenta, C. Waga, and C. Valladares, *An integer programming approach to equitable coloring problems*, in Proceedings of the XXXIX Brazilian Symposium on Operations Research, vol. 1, Fortaleza, 2007, pp. 1795–1801.

[2] Campêlo, M., V. Campos, and R. Corrêa, *On the asymmetric representatives formulation for the vertex coloring problem*, Electronic Notes in Discrete Mathematics **19** (2005) pp. 337–343.

[3] Campêlo, M., R. C. Corrêa, and Y. Frota, *Cliques, holes and the vertex coloring polytope*, Information Processing Letters **89** (2004) pp. 159–164.

[4] Frota, Y., N. Maculan, T. F. Noronha, and C.C. Ribeiro, *A branch-and-cut algorithm for partition coloring*, Networks, 2009, to appear.

[5] Furmańczyk, H., and M. Kubale, *The Complexity of Equitable Vertex Coloring of Graphs*, Journal of Applied Computer Science **13** (2005) pp. 95–107.

[6] Méndez-Díaz, I., G. Nasini, and D. Severin, *A polyhedral approach for the graph equitable coloring problem*, in Proceedings of the VI ALIO/EURO Workshop on Applied Combinatorial Optimization, Buenos Aires, 2008.

[7] Meyer, W., *Equitable coloring*, American Mathematical Monthly **80** (1973) pp. 143–149.

[8] Touhami, S., *Optimization problems in cellular Networks*, PhD thesis, John Molson School of Business, Concordia University, Montreal, 2004.

[9] Tucker, A., *Perfect graphs and an application to optimizing municipal services*, SIAM Review **15** (1973) pp. 585–590.